```
FFFFFFFFFF   000000    RRRRRRRR    IIIIII    000000   BBBBBBBB   EEEEEEEEEE   GGGGGGGG
FFFFFFFFFF   000000    RRRRRRRR    IIIIII    000000   BBBBBBBB   EEEEEEEEEE   GGGGGGGG
FF           00    00  RR      RR     II    00    00  BB     BB  EE           GG
FF           00    00  RR      RR     II    00    00  BB     BB  EE           GG
FF           00    00  RR      RR     II    00    00  BB     BB  EE           GG
FFFFFFFF     00    00  RRRRRRRR       II    00    00  BBBBBBBB   EEEEEEEE      GG
FFFFFFFF     00    00  RRRRRRRR       II    00    00  BBBBBBBB   EEEEEEEE      GG
FF           00    00  RR   RR        II    00    00  BB     BB  EE           GG   GGGGGG
FF           00    00  RR    RR       II    00    00  BB     BB  EE           GG   GGGGGG
FF           00    00  RR     RR      II    00    00  BB     BB  EE           GG      GG
FF           00    00  RR     RR      II    00    00  BB     BB  EE           GG      GG
FF           000000    RR      RR   IIIIII  000000   BBBBBBBB   EEEEEEEEEE    GGGGGG      ....
FF           000000    RR      RR   IIIIII  000000   BBBBBBBB   EEEEEEEEEE    GGGGGG      ....
```

```
LL           IIIIII     SSSSSSSS
LL           IIIIII     SSSSSSSS
LL             II     SS
LL             II     SS
LL             II     SS
LL             II       SSSSSS
LL             II       SSSSSS
LL             II             SS
LL             II             SS
LL             II             SS
LL             II             SS
LLLLLLLLLL   IIIIII    SSSSSSSS
LLLLLLLLLL   IIIIII    SSSSSSSS
```

```
   1    0001   0 MODULE FOR$$IO_BEG (%TITLE'FORTRAN READ/WRITE statement initialization'
   2    0002   0            IDENT = '2-006'              ! File: FORIOBEG.B32  Edit: SBL2006
   3    0003   0            ) =
   4    0004   1 BEGIN
   5    0005   1
   6    0006   1 !****************************************************************
   7    0007   1 !*                                                              *
   8    0008   1 !*    COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                   *
   9    0009   1 !*    DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.    *
  10    0010   1 !*    ALL RIGHTS RESERVED.                                      *
  11    0011   1 !*                                                              *
  12    0012   1 !*    THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED *
  13    0013   1 !*    ONLY IN  ACCORDANCE WITH  THE  TERMS OF  SUCH  LICENSE  AND WITH THE  *
  14    0014   1 !*    INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER *
  15    0015   1 !*    COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY *
  16    0016   1 !*    OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY *
  17    0017   1 !*    TRANSFERRED.                                              *
  18    0018   1 !*                                                              *
  19    0019   1 !*    THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE *
  20    0020   1 !*    AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT *
  21    0021   1 !*    CORPORATION.                                              *
  22    0022   1 !*                                                              *
  23    0023   1 !*    DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS *
  24    0024   1 !*    SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.   *
  25    0025   1 !*                                                              *
  26    0026   1 !*                                                              *
  27    0027   1 !****************************************************************
  28    0028   1
  29    0029   1 !++
  30    0030   1 ! FACILITY: FORTRAN Support Library - Not user callable
  31    0031   1 !
  32    0032   1 ! ABSTRACT:
  33    0033   1 !
  34    0034   1 !     This module contains the common initialization code for
  35    0035   1 !     all FORTRAN multi-call I/O statements (READ, WRITE,
  36    0036   1 !     ENCODE, DECODE, REWRITE, PRINT, TYPE and ACCEPT).
  37    0037   1 !
  38    0038   1 ! ENVIRONMENT: User access mode; mixture of AST level or not.
  39    0039   1 !
  40    0040   1 ! AUTHOR: Thomas N. Hastings, CREATION DATE: 08-Mar-77: Version 01
  41    0041   1 !     Steven B. Lionel, 4-Dec-1979: Version 2
  42    0042   1 !
  43    0043   1 !
  44    0044   1 ! 2-001 - All new logic, optimized for high speed.  Steve Lionel
  45    0045   1 !         with many helpful suggestions from Rich Grove.  4-Dec-1979
  46    0046   1 ! 2-002 - Fixed bug in run-time formatting.  SBL 11-Dec-1979
  47    0047   1 ! ***** - VMS V2.0
  48    0048   1 ! 2-003 - Add support for NAMELIST.  Also move BUILTIN declaration of
  49    0049   1 !         ACTUALCOUNT to inside the routine which uses it.  SBL 21-August-1980
  50    0050   1 ! 2-004 - Declare ARGS to be 4 bytes since the second byte is looked at.
  51    0051   1 !         BLISS V2.0 didn't catch it, but V2.1 did.  SBL 14-Oct-1980
  52    0052   1 ! 2-005 - Enhance MIXFILACC message.  JAW 22-Aug-1981
  53    0053   1 ! ***** - VMS V3.0
  54    0054   1 ! 2-006 - Add list-directed internal files.  Use prologue file.  SBL 21-Apr-1983
  55    0055   1 !--
  56    0056   1
```

```
 58         0057  1 !
 59         0058  1 ! PROLOGUE FILE:
 60         0059  1 !
 61         0060  1
 62         0061  1 REQUIRE 'RTLIN:FORPROLOG';                         ! FORTRAN definitions
 63         0127  1 SWITCHES ZIP;                                      ! Optimize for speed
 64         0128  1
 65         0129  1 !
 66         0130  1 ! TABLE OF CONTENTS:
 67         0131  1 !
 68         0132  1
 69         0133  1 FORWARD ROUTINE
 70         0134  1     FOR$$IO_BEG : CALL_FIOBEG NOVALUE;             ! Common routine for all
 71         0135  1
 72         0136  1 !
 73         0137  1 ! MACROS:
 74         0138  1 !
 75         0139  1
 76         0140  1 MACRO
 77         0141  1     POS (A) = %FIELDEXPAND(A,1) %,          ! Gets bit position from LUB$V symbol
 78         0142  1
 79         0143  1     MASK (A) = 1^POS(A) %;                  ! Mask for LUB$V symbol
 80         0144  1
 81         0145  1 !
 82         0146  1 ! EQUATED SYMBOLS:
 83         0147  1 !
 84         0148  1 LITERAL
 85         0149  1
 86         0150  1     !+
 87         0151  1     ! Masks for denoting which arguments are present for each statement
 88         0152  1     ! type.  The two M_TST_ masks are used for testing combined attributes
 89         0153  1     ! of a statement type.
 90         0154  1     !-
 91         0155  1     M_ARG_FMT   = 1^0,              ! 1 if format is present
 92         0156  1     M_ARG_REC   = 1^1,              ! 1 if record number is present
 93         0157  1     M_ARG_USR   = 1^2,              ! 1 if user buffer is present
 94         0158  1     M_ARG_KEY   = 1^3,              ! 1 if key fields are present
 95         0159  1     M_TST_INT   = 1^4,              ! 1 if internal file or ENCODE/DECODE
 96         0160  1     M_TST_FMT   = 1^5,              ! 1 if formatted or list-directed
 97         0161  1
 98         0162  1     !+
 99         0163  1     ! Masks which select which unit attributes are NOT allowed for
100         0164  1     ! a statement type.
101         0165  1     !-
102         0166  1     M_ATR_RON   = MASK (LUB$V_READ_ONLY),          ! 1 if READ_ONLY prohibited
103         0167  1     M_ATR_DIR   = MASK (LUB$V_DIRECT),             ! 1 if DIRECT prohibited
104         0168  1     M_ATR_FMT   = MASK (LUB$V_FORMATTED),          ! 1 if FORMATTED prohibited
105         0169  1     M_ATR_UNF   = MASK (LUB$V_UNFORMAT),           ! 1 if UNFORMATTED prohibited
106         0170  1     M_ATR_SEQ   = MASK (LUB$V_SEQUENTIA),          ! 1 if SEQUENTIAL prohibited
107         0171  1     M_ATR_KEY   = MASK (LUB$V_KEYED);              ! 1 if KEYED prohibited
108         0172  1
109         0173  1     !
110         0174  1     ! FIELD DECLARATIONS:
111         0175  1     !
112         0176  1 FIELD
113         0177  1
114         0178  1     DUMMY_FIELDS =
```

```
 115     0179  1                       !+
 116     0180  1                       ! The purpose of this fieldset is only to define the field
 117     0181  1                       ! FOR_V_OBJ_FMT so that it can be used for TST_OBJ below.
 118     0182  1                       !-
 119     0183  1                       SET
 120     0184  1                       FOR_V_OBJ_FMT = [FOR$V_OBJ_FMT]
 121     0185  1                       TES;
 122     0186  1
 123     0187  1                   ARG_FIELDS =
 124     0188  1                       !+
 125     0189  1                       ! See definition of M_ARG_x and M_TST_x literals above.
 126     0190  1                       !-
 127     0191  1                       SET
 128     0192  1                       ARG_FMT = [0,0,1,0],
 129     0193  1                       ARG_REC = [0,1,1,0],
 130     0194  1                       ARG_USR = [0,2,1,0],
 131     0195  1                       ARG_KEY = [0,3,1,0],
 132     0196  1                       TST_INT = [0,4,1,0],
 133     0197  1                       TST_FMT = [0,5,1,0],
 134     0198  1                       TST_OBJ = [0,POS (FOR_V_OBJ_FMT),1,0]   ! 1 if run-time format
 135     0199  1                       TES;
 136     0200  1
 137     0201  1                   ATR_FIELDS =
 138     0202  1                       !+
 139     0203  1                       ! See definition of M_ATR_x literals above.
 140     0204  1                       !-
 141     0205  1                       SET
 142     0206  1                       ATR_RON = [0,POS (LUB$V_READ_ONLY),1,0],
 143     0207  1                       ATR_DIR = [0,POS (LUB$V_DIRECT),1,0],
 144     0208  1                       ATR_FMT = [0,POS (LUB$V_FORMATTED),1,0],
 145     0209  1                       ATR_UNF = [0,POS (LUB$V_UNFORMAT),1,0],
 146     0210  1                       ATR_SEQ = [0,POS (LUB$V_SEQUENTIA),1,0],
 147     0211  1                       ATR_KEY = [0,POS (LUB$V_KEYED),1,0]
 148     0212  1                       TES;
 149     0213  1
 150     0214  1           !
 151     0215  1           ! OWN STORAGE:
 152     0216  1           !
 153     0217  1
 154     0218  1
 155     0219  1   BIND
 156     0220  1           ERR_ADR_IDX =               ! For each statement type, gives
 157     0221  1                                       ! the argument list index for the
 158     0222  1                                       ! ERR= parameter.  Numbering starts
 159     0223  1                                       ! at 1.
 160     0224  1               UPLIT BYTE (
 161     0225  1
 162     0226  1                       0,                                          ! unused
 163     0227  1                       3,                                          ! WRITE sequential formatted
 164     0228  1                       3,                                          ! READ sequential formatted
 165     0229  1                       2,                                          ! WRITE sequential unformatted
 166     0230  1                       2,                                          ! READ sequential unformatted
 167     0231  1                       4,                                          ! WRITE direct formatted
 168     0232  1                       4,                                          ! READ direct formatted
 169     0233  1                       3,                                          ! WRITE direct unformatted
 170     0234  1                       3,                                          ! READ direct unformatted
 171     0235  1                       2,                                          ! WRITE sequential list-directed
```

```
 172     0236  1                          2,                                      ! READ sequential list-directed
 173     0237  1                          4,                                      ! ENCODE formatted
 174     0238  1                          4,                                      ! DECODE formatted
 175     0239  1                          3,                                      ! REWRITE formatted
 176     0240  1                          6,                                      ! READ keyed formatted
 177     0241  1                          2,                                      ! REWRITE unformatted
 178     0242  1                          5,                                      ! READ keyed unformatted
 179     0243  1                          3,                                      ! WRITE internal formatted
 180     0244  1                          3,                                      ! READ internal formatted
 181     0245  1                          3,                                      ! WRITE sequential NAMELIST
 182     0246  1                          3,                                      ! READ sequential NAMELIST
 183     0247  1                          2,                                      ! WRITE internal list-directed
 184     0248  1                          2                                       ! READ internal list-directed
 185     0249  1
 186     0250  1                  ) : VECTOR [ISB$K_FORSTTYHI+1, BYTE],
 187     0251  1
 188     0252  1          STMT_ARG =                                      ! A table indexed by statement type
 189     0253  1                                                          ! that has a bit set in the appropriate
 190     0254  1                                                          ! position if an argument is defined
 191     0255  1                                                          ! for that statement.  Other bits
 192     0256  1                                                          ! are used for combined tests.
 193     0257  1                                                          ! See above for literal definitions.
 194     0258  1              UPLIT BYTE (
 195     0259  1
 196     0260  1                  0,                                              ! unused
 197     0261  1                  M_ARG_FMT+M_TST_FMT,                            ! WRITE sequential formatted
 198     0262  1                  M_ARG_FMT+M_TST_FMT,                            ! READ sequential formatted
 199     0263  1                  0,                                              ! WRITE sequential unformatted
 200     0264  1                  0,                                              ! READ sequential unformatted
 201     0265  1                  M_ARG_FMT+M_ARG_REC+M_TST_FMT,                  ! WRITE direct formatted
 202     0266  1                  M_ARG_FMT+M_ARG_REC+M_TST_FMT,                  ! READ direct formatted
 203     0267  1                  M_ARG_REC,                                      ! WRITE direct unformatted
 204     0268  1                  M_ARG_REC,                                      ! READ direct unformatted
 205     0269  1                  M_TST_FMT,                                      ! WRITE sequential list-directed
 206     0270  1                  M_TST_FMT,                                      ! READ sequential list-directed
 207     0271  1                  M_ARG_FMT+M_ARG_USR+M_TST_INT+M_TST_FMT,        ! ENCODE formatted
 208     0272  1                  M_ARG_FMT+M_ARG_USR+M_TST_INT+M_TST_FMT,        ! DECODE formatted
 209     0273  1                  M_ARG_FMT+M_TST_FMT,                            ! REWRITE formatted
 210     0274  1                  M_ARG_FMT+M_ARG_KEY+M_TST_FMT,                  ! READ keyed formatted
 211     0275  1                  0,                                              ! REWRITE unformatted
 212     0276  1                  M_ARG_KEY,                                      ! READ keyed unformatted
 213     0277  1                  M_ARG_FMT+M_TST_INT+M_TST_FMT,                  ! WRITE internal formatted
 214     0278  1                  M_ARG_FMT+M_TST_INT+M_TST_FMT,                  ! READ internal formatted
 215     0279  1                  M_ARG_FMT+M_TST_FMT,                            ! WRITE sequential NAMELIST
 216     0280  1                  M_ARG_FMT+M_TST_FMT,                            ! READ sequential NAMELIST
 217     0281  1                  M_TST_INT+M_TST_FMT,                            ! WRITE internal list-directed
 218     0282  1                  M_TST_INT+M_TST_FMT                             ! READ internal list-directed
 219     0283  1
 220     0284  1                  ) : VECTOR [ISB$K_FORSTTYHI+1,BYTE],
 221     0285  1
 222     0286  1          STMT_ATR =
 223     0287  1                                                          ! A table of statement
 224     0288  1                                                          ! attributes indexed by
 225     0289  1                                                          ! statement type.  If a
 226     0290  1                                                          ! bit is set, the corresponding
 227     0291  1                                                          ! attribute is NOT permitted
 228     0292  1                                                          ! to be defined for the unit.
```

```
229   0293   1                                                          ! A table entry is ANDed
230   0294   1                                                          ! with LUB$W_UNIT_ATTR.  If
231   0295   1                                                          ! the result is non-zero,
232   0296   1                                                          ! there is a disallowed
233   0297   1                                                          ! combination.
234   0298   1              UPLIT WORD (
235   0299   1
236   0300   1                  0,                                                       ! unused
237   0301   1                  M_ATR_RON+M_ATR_DIR+M_ATR_UNF,                           ! WRITE sequential formatted
238   0302   1                  M_ATR_DIR+M_ATR_UNF,                                     ! READ sequential formatted
239   0303   1                  M_ATR_RON+M_ATR_DIR+M_ATR_FMT,                           ! WRITE sequential unformatted
240   0304   1                  M_ATR_DIR+M_ATR_FMT,                                     ! READ sequential unformatted
241   0305   1                  M_ATR_RON+M_ATR_SEQ+M_ATR_KEY+M_ATR_UNF,                 ! WRITE direct formatted
242   0306   1                  M_ATR_SEQ+M_ATR_KEY+M_ATR_UNF,                           ! READ direct formatted
243   0307   1                  M_ATR_RON+M_ATR_SEQ+M_ATR_KEY+M_ATR_FMT,                 ! WRITE direct unformatted
244   0308   1                  M_ATR_SEQ+M_ATR_KEY+M_ATR_FMT,                           ! READ direct unformatted
245   0309   1                  M_ATR_RON+M_ATR_DIR+M_ATR_KEY+M_ATR_UNF,                 ! WRITE sequential list-directed
246   0310   1                  M_ATR_DIR+M_ATR_KEY+M_ATR_UNF,                           ! READ sequential list-directed
247   0311   1                  0,                                                       ! ENCODE formatted
248   0312   1                  0,                                                       ! DECODE formatted
249   0313   1                  M_ATR_RON+M_ATR_DIR+M_ATR_UNF,                           ! REWRITE formatted
250   0314   1                  M_ATR_DIR+M_ATR_SEQ+M_ATR_UNF,                           ! READ keyed formatted
251   0315   1                  M_ATR_RON+M_ATR_DIR+M_ATR_FMT,                           ! REWRITE unformatted
252   0316   1                  M_ATR_DIR+M_ATR_SEQ+M_ATR_FMT,                           ! READ keyed unformatted
253   0317   1                  0,                                                       ! WRITE internal formatted
254   0318   1                  0,                                                       ! READ internal formatted
255   0319   1                  M_ATR_RON+M_ATR_DIR+M_ATR_UNF,                           ! WRITE sequential NAMELIST
256   0320   1                  M_ATR_DIR+M_ATR_UNF,                                     ! READ sequential NAMELIST
257   0321   1                  0,                                                       ! WRITE internal list-directed
258   0322   1                  0                                                        ! READ internal list-directed
259   0323   1
260   0324   1              ) : VECTOR [ISB$K_FORSTTYHI+1, WORD];
261   0325   1
262   0326   1  !
263   0327   1  !
264   0328   1  !  EXTERNAL REFERENCES:
265   0329   1  !
266   0330   1  !
267   0331   1
268   0332   1  EXTERNAL ROUTINE
269   0333   1      FOR$$CB_PUSH : JSB_CB_PUSH NOVALUE,                   ! Create LUB/ISB/RAB, if needed, for this unit and
270   0334   1                                                           ! push down I/O system
271   0335   1      FOR$$FMT_COMPIL : WEAK,                               ! Format compiler - returns adr
272   0336   1                                                           ! of compiled format
273   0337   1      FOR$$ERR_ENDHND,                                     ! error condition handler for ERR= and END=
274   0338   1      FOR$$SIGNAL_STO : NOVALUE,                           ! Convert FORTRAN  err#
275   0339   1                                                           ! to VAX error # and SIGNAL_STOP.
276   0340   1      FOR$$OPEN_DEFLT : CALL_CCB NOVALUE;                  ! Perform default OPEN
277   0341   1
278   0342   1  EXTERNAL
279   0343   1      FOR$$AA_UDF_PRO : VECTOR,                            ! UDF level initialization dispatch
280   0344   1                                                           ! table
281   0345   1      FOR$$IO_IN_PROG;                                     ! I/O in progress handler
282   0346   1
```

```
 284      0347  1  GLOBAL ROUTINE FOR$$IO_BEG (FLAGS_ARG, UNIT) : CALL_FIOBEG NOVALUE =
 285      0348  1  !+
 286      0349  1  !  FUNCTIONAL DESCRIPTION:
 287      0350  1  !
 288      0351  1  !     Common I/O statement initialization:
 289      0352  1  !
 290      0353  1  !     1. Determine if ERR= and/or END= optional parameters
 291      0354  1  !        are present or not.
 292      0355  1  !     2. Setup an error handler.
 293      0356  1  !     3. Setup a LUB/ISB/RAB control block for this logical unit
 294      0357  1  !        if not setup already.
 295      0358  1  !     4. Check for incorrect mixing of I/O statements.
 296      0359  1  !     5. If unit not already OPEN, OPEN it.
 297      0360  1  !     6. Store away passed parameters.
 298      0361  1  !
 299      0362  1  !  FORMAL PARAMETERS:
 300      0363  1  !
 301      0364  1  !     FLAGS_ARG.rl.v  - This contains the statement type number is
 302      0365  1  !                       bits 0:7.  If bit FOR$V_OBJ_FMT (8) is set,
 303      0366  1  !                       then this is a run-time (historically "object-time")
 304      0367  1  !                       formatted statement and requires the format
 305      0368  1  !                       compiler to be defined (via a strong .EXTERN
 306      0369  1  !                       somewhere else).
 307      0370  1  !                       This parameter is passed in R0.
 308      0371  1  !
 309      0372  1  !     UNIT            - The first of the arguments pointed to by the
 310      0373  1  !                       AP.  This is the unit number passed by value
 311      0374  1  !                       except for the following statements:
 312      0375  1  !                          ENCODE,DECODE  - byte count
 313      0376  1  !                          Internal file  - address of user
 314      0377  1  !                                           variable descriptor.
 315      0378  1  !                       Remaining arguments are selected based on the
 316      0379  1  !                       ARG_TYPES vector of bits.
 317      0380  1  !
 318      0381  1  !  IMPLICIT INPUTS:
 319      0382  1  !
 320      0383  1  !     LUB$V_SEQUENTIA               This unit has been specified for
 321      0384  1  !                                   sequential access by a previous OPEN.
 322      0385  1  !     LUB$V_DIRECT                  This unit has been specified for
 323      0386  1  !                                   direct access by a previous OPEN
 324      0387  1  !                                   or DEFINE FILE.
 325      0388  1  !     LUB$V_KEYED                   This unit has been specified for
 326      0389  1  !                                   keyed access by a previous OPEN.
 327      0390  1  !     LUB$V_FORMATTED               This unit has been specified for
 328      0391  1  !                                   formatted I/O by a previous OPEN
 329      0392  1  !                                   or default OPEN.
 330      0393  1  !     LUB$V_UNFORMAT                This unit has been specified for
 331      0394  1  !                                   unformatted I/O by a previous
 332      0395  1  !                                   OPEN, DEFINE FILE, or default OPEN.
 333      0396  1  !     LUB$V_READ_ONLY               This unit has been specified for
 334      0397  1  !                                   performing READs only by the current
 335      0398  1  !                                   OPEN or CALL FDBSET.
 336      0399  1  !     LUB$V_OPENED                  This unit has been opened by a previous
 337      0400  1  !                                   OPEN, or default OPEN (for READ/WRITE
 338      0401  1  !                                   OR ENDFILE).
 339      0402  1  !
 340      0403  1  !  IMPLICIT OUTPUTS:
```

FOR$$IO_BEG   FORTRAN READ/WRITE statement initialization   16-Sep-1984 00:29:21   VAX-11 Bliss-32 V4.0-742   Page 7
2-006   14-Sep-1984 12:32:03   [FORRTL.SRC]FORIOBEG.B32;1   (3)

K 4

```
 341    0404  1 !        LUB$L_LOG_RECNO                    Current logical (or spanned)
 342    0405  1 !                                           record number for sequential access
 343    0406  1 !                                           files (needed for BACKSPACE of spanned
 344    0407  1 !                                           records). Current FORTRAN direct
 345    0408  1 !                                           access files 1 = first record.
 346    0409  1 !                                           0 never stored.
 347    0410  1 !
 348    0411  1 !        ISB$A_ERR_EQUAL                    Adr. of jump to if error occurs
 349    0412  1 !                                           (ERR= supplied) or 0
 350    0413  1 !        ISB$A_END_EQUAL                    Adr. to jump to if end of file
 351    0414  1 !                                           occurs (END= supplied) or 0.
 352    0415  1 !        ISB$B_ERR_NO                       0.  Last continuable error during statement
 353    0416  1 !        ISB$A_FMT_BEG                      If object-time format, Adr. of first
 354    0417  1 !                                           char in resultant format array.
 355    0418  1 !        RAB$B_KRF                          set to keyid if present and not -1
 356    0419  1 !        RAB$V_KGE                          set if match present and is 1
 357    0420  1 !        RAB$V_KGT                          set if match present and is 2
 358    0421  1 !        RAB$L_KBF                          set to the key address
 359    0422  1 !        RAB$B_KSZ                          set to to key size or zero if not string
 360    0423  1 !
 361    0424  1 ! ROUTINE VALUE:
 362    0425  1 !
 363    0426  1 !     NONE
 364    0427  1 !
 365    0428  1 ! SIDE EFFECTS:
 366    0429  1 !
 367    0430  1 !     Allocates a LUB/ISB/RAB block if necessary.
 368    0431  1 !     Initiates activity on an ISB.
 369    0432  1 !     Opens a unit if necessary.
 370    0433  1 !
 371    0434  1 ! NOTES:
 372    0435  1 !     In the Run-Time Library, FOR$$IO_BEG is never actually called.
 373    0436  1 !     Each statement type has its own entry point which places the
 374    0437  1 !     correct type number in R0 and then branches to the FOR$$IO_BEG+2.
 375    0438  1 !     These separate entry points also make the required external
 376    0439  1 !     references to the UDF and REC level routines and the format
 377    0440  1 !     compiler if necessary.
 378    0441  1 !--
 379    0442  1
 380    0443  2     BEGIN
 381    0444  2
 382    0445  2     GLOBAL REGISTER
 383    0446  2         CCB = K_CCB_REG : REF $FOR$CCB_DECL;
 384    0447  2
 385    0448  2     BUILTIN
 386    0449  2         ACTUALCOUNT,                       ! The number of arguments we were called with
 387    0450  2         FP,                                ! Our frame pointer
 388    0451  2         AP;                                ! Reference to the "caller" argument list
 389    0452  2
 390    0453  2     LOCAL                                  ! The first 4 locals are used by error-processing routines:
 391    0454  2         L_UNWIND_ACTION : VOLATILE,        ! Unwind action code (FOR$K_UNWIND{POP or NOP}
 392    0455  2         A_ERR_ADR : VOLATILE,              ! User-program supplied ERR= address  (0 if none)
 393    0456  2         A_END_ADR : VOLATILE,              ! User-program supplied END= address  (0 if none)
 394    0457  2         L_UNWIND_DEPTH : VOLATILE,         ! No. of additional frames to unwind if error
 395    0458  2         ! produced at compiled time or object time
 396    0459  2         STMT_TYPE,                         ! Statement type number
 397    0460  2         ERR_POS : REF VECTOR [,LONG],      ! Address of err_adr parameter
```

```
398    0461   2              ARGS : BLOCK [4, BYTE] FIELD (ARG_FIELDS),        ! Argument flags
399    0462   2              PTR : REF VECTOR [,LONG];                         ! Argument list pointer
400    0463   2
401    0464   2          STACKLOCAL
402    0465   2              ARG_LIST_END;                                     ! Address of last actual argument
403    0466   2
404    0467   2          MAP
405    0468   2              FLAGS_ARG : BLOCK [4,BYTE],                       ! Passed in R0
406    0469   2              AP : REF VECTOR [,LONG],                          ! Pointer to argument list
407    0470   2              FP : REF BLOCK [,BYTE];
408    0471   2
409    0472   2          ENABLE                                               ! Establish error handler and provide arguments:
410    0473   2              !    UNWIND action code, depth to unwind (0)
411    0474   2              !    ERR= and END= addresses from caller
412    0475   2              FOR$$ERR_ENDHND (L_UNWIND_ACTION, A_ERR_ADR, A_END_ADR, L_UNWIND_DEPTH);
413    0476   2
414    0477   2          !+
415    0478   2          ! Copy flags argument passed by ''caller'' in R0
416    0479   2          !-
417    0480   2
418    0481   2          !+
419    0482   2          ! Set STMT_TYPE to FORTRAN statement type. Set up ARGS with bit
420    0483   2          ! for run-time formatting.
421    0484   2          !-
422    0485   2          STMT_TYPE = .FLAGS_ARG [FOR$B_STMT_TYPE];
423    0486   2          FLAGS_ARG [FOR$B_STMT_TYPE] = 0;
424    0487   2          ARGS = .STMT_ARG [.STMT_TYPE] OR .FLAGS_ARG;
425    0488   2
426    0489   2          !+
427    0490   2          ! Set cleanup action on UNWIND to no-operation (since
428    0491   2          ! LUB/ISB/RAB not pushed down yet).
429    0492   2          ! Also set L_UNWIND_DEPTH to additional no. of stack frames between
430    0493   2          ! establisher and user program to be unwound in order to
431    0494   2          ! get back to user program.
432    0495   2          !-
433    0496   2
434    0497   2          L_UNWIND_ACTION = FOR$K_UNWINDNOP;
435    0498   2
436    0499   2          !+
437    0500   2          ! Setup LOCAL A_ERR_ADR and A_END_ADR to pass to error handler
438    0501   2          ! in case of a SIGNAL.
439    0502   2          !-
440    0503   2
441    0504   2          ARG_LIST_END = AP [ACTUALCOUNT ()];              ! Get address of last entry
442    0505   2          ERR_POS = AP [.ERR_ADR_IDX [.STMT_TYPE]];
443    0506   2          IF .ARG_LIST_END GEQA ERR_POS [0]
444    0507   2          THEN
445    0508   3              BEGIN
446    0509   3              IF .ARG_LIST_END GTRA ERR_POS [0]
447    0510   3              THEN
448    0511   3                  A_END_ADR = .ERR_POS [1];
449    0512   3              A_ERR_ADR = .ERR_POS [0];
450    0513   2              END;
451    0514   2
452    0515   2
453    0516   2
454    0517   2          !+
```

```
 455    0518   2     ! Call FOR$$CB_PUSH to initiate I/O on this unit.  If this is
 456    0519   2     ! an internal file I/O or ENCODE/DECODE, then use a special
 457    0520   2     ! logical unit number.
 458    0521   2     !-
 459    0522
 460    0523   3     IF NOT .ARGS [TST_INT]                    ! Not internal file type
 461    0524         THEN
 462    0525   2         FOR$$CB_PUSH (.UNIT, LUB$K_DLUN_MIN)
 463    0526         ELSE
 464    0527   2         FOR$$CB_PUSH (LUB$K_LUN_ENCD, LUB$K_LUN_ENCD);
 465    0528   2
 466    0529   2     L_UNWIND_ACTION = FOR$K_UNWINDPOP;
 467    0530
 468    0531         !+
 469    0532   2     ! Store away ERR= and END= address for duration of I/O
 470    0533         ! statement.
 471    0534         ! Store I/O statement type code for
 472    0535   2     ! future dispatching to other levels of abstraction during
 473    0536         ! this I/O statement.
 474    0537   2     ! Clear last continuable error byte in ISB.
 475    0538   2     !-
 476    0539
 477    0540   2     CCB [ISB$A_ERR_EQUAL] = .A_ERR_ADR;
 478    0541   2     CCB [ISB$A_END_EQUAL] = .A_END_ADR;
 479    0542   2     CCB [ISB$B_ERR_NO] = 0;
 480    0543   2     CCB [ISB$B_STTM_TYPE] = .STMT_TYPE;
 481    0544   2
 482    0545   2     !+
 483    0546         ! Check for the following errors:
 484    0547         !   OPEN or DEFINE FILE required for keyed or direct access
 485    0548   2     !   mixed file access modes
 486    0549         !   write to READONLY file
 487    0550         ! This is done by ANDing the word in the LUB that has unit attribute
 488    0551   2     ! bits with the appropriate mask in STMT_ATR.  If any bit is still on,
 489    0552   2     ! then at least one invalid combination was detected.  The bits are
 490    0553   2     ! then analyzed to determine which error was found.
 491    0554   2     !-
 492    0555
 493    0556   2     IF (.STMT_ATR [.STMT_TYPE] AND .CCB [LUB$W_UNIT_ATTR]) NEQ 0
 494    0557   2     THEN
 495    0558   3         BEGIN
 496    0559   3         !+
 497    0560         ! If we get here, then we know there is an invalid combination.
 498    0561         ! Give the appropriate error message depending on which bit
 499    0562   3     ! is still on.
 500    0563   3     !-
 501    0564         LOCAL
 502    0565   3         ATTR : BLOCK [1,WORD] FIELD (ATR_FIELDS);
 503    0566         !+
 504    0567         ! The following assignment is done in two statements to prevent
 505    0568         ! BLISS from making a common subexpression with the above test.
 506    0569         !-
 507    0570   3     ATTR = .STMT_ATR [.STMT_TYPE];
 508    0571   3     ATTR = .ATTR AND .CCB [LUB$W_UNIT_ATTR];
 509    0572   3     IF .ATTR [ATR_SEQ]
 510    0573   3     THEN
 511    0574   4         BEGIN           ! Can't be ACCESS='SEQUENTIAL'
```

```
  512      0575   4                           FOR$$SIGNAL_STO (FOR$K_OPEDEFREQ);
  513      0576   4                           RETURN;
  514      0577   3                           END;
  515      0578   3                   IF .ATTR [ATR_RON]
  516      0579   3                   THEN
  517      0580   4                       BEGIN          ! Can't be READONLY
  518      0581   4                       FOR$$SIGNAL_STO (FOR$K_WRIREAFIL);
  519      0582   4                       RETURN;
  520      0583   3                       END;
  521      0584   3                   !+
  522      0585   3                   ! If it isn't either of the above, then it must be mixed access
  523      0586   3                   ! modes or formatting types.  Signal MIXFILACC as the primary
  524      0587   3                   ! message, with explanatory chained message.  Note that direct
  525      0588   3                   ! or keyed I/O to a sequential unit has already been rejected
  526      0589   3                   ! above with OPEDEFREQ.
  527      0590   3                   !-
  528      0591   3                   FOR$$SIGNAL_STO (FOR$K_MIXFILACC,
  529      0592   3                               !+
  530      0593   3                               ! Choose the appropriate secondary message.
  531      0594   3                               !-
  532      0595   3                               IF      .ATTR [ATR_UNF] THEN FOR$_FMTIO_UNF
  533      0596   3                               ELSE IF .ATTR [ATR_FMT] THEN FOR$_UNFIO_FMT
  534      0597   3                               ELSE IF .ATTR [ATR_KEY] THEN FOR$_DIRIO_KEY
  535      0598   3                               ELSE IF .ATTR [ATR_DIR] THEN
  536      0599   3                                   IF .ARGS [ARG_KEY]              ! Check statement type
  537      0600   3                                       THEN FOR$_KEYIO_DIR
  538      0601   3                                       ELSE FOR$_SEQIO_DIR
  539      0602   3                                   ELSE 0
  540      0603   3                           );
  541      0604   3                   RETURN;
  542      0605   2                   END;
  543      0606   2
  544      0607   2           !+
  545      0608   2           ! We now start picking up arguments from the argument list.  PTR
  546      0609   2           ! will be the pointer to the current place in the argument list.
  547      0610   2           ! Depending on bits set in ARGS, arguments will be taken and
  548      0611   2           ! PTR advanced.
  549      0612   2           !-
  550      0613   2
  551      0614   2           PTR = AP [2];                    ! Start with second argument
  552      0615   2
  553      0616   2           !+
  554      0617   2           ! Get record number if present
  555      0618   2           !-
  556      0619   2
  557      0620   2           IF .ARGS [ARG_REC]
  558      0621   2           THEN
  559      0622   3               BEGIN
  560      0623   3               IF .PTR [0] EQL 0 OR
  561      0624   4                   (.CCB [LUB$L_REC_MAX] NEQ 0 AND (.PTR [0] GTRU .CCB [LUB$L_REC_MAX]))
  562      0625   3               THEN
  563      0626   3                   !+
  564      0627   3                   ! The record number was zero or was greater than the
  565      0628   3                   ! maximum for this file.
  566      0629   3                   !-
  567      0630   4                   BEGIN
  568      0631   4                   FOR$$SIGNAL_STO (FOR$K_RECNUMOUT);
```

```
569     0632   4                        RETURN;
570     0633   4                        END;
571     0634   3                   CCB [LUB$L_LOG_RECNO] = RLONG_A (PTR);   ! Pick up logical record number
572     0635   3                   END;
573     0636   2
574     0637   2            !+
575     0638   2            ! If this is a run-time (object-time) format,
576     0639   2            ! compile format and store address and length in ISB.
577     0640   2            ! Otherwise store the address of the pre-compiled format into the ISB.
578     0641   2            ! Note: a NAMELIST description block is passed as if were a compiled
579     0642   2            ! format, so it is stored here.
580     0643   2            !-
581     0644   2
582     0645   2            IF .ARGS [ARG_FMT]
583     0646   2            THEN
584     0647   2                IF NOT .ARGS [TST_OBJ]
585     0648   2                THEN
586     0649   3                    CCB [ISB$A_FMT_BEG] = RLONG_A (PTR)
587     0650   2                ELSE
588     0651   2                    FOR$$FMT_COMPIL (RLONG_A (PTR), CCB [ISB$W_FMT_LEN], CCB [ISB$A_FMT_BEG]);
589     0652   2
590     0653   2
591     0654   2            !+
592     0655   2            ! If the unit is open, check to see if it was opened by ENDFILE.
593     0656   2            ! If it was, complete the attribute specifications based on the
594     0657   2            ! statement type.
595     0658   2            ! If the unit is not open, open it using default attributes based
596     0659   2            ! on the statement type.
597     0660   2            !-
598     0661   2
599     0662   2            IF .CCB [LUB$V_OPENED]                     ! Unit opened
600     0663   2            THEN
601     0664   3                BEGIN
602     0665   3                IF .CCB [LUB$V_ENDFILOPN]              ! Opened by ENDFILE
603     0666   3                THEN
604     0667   4                    BEGIN
605     0668   4                    CCB [LUB$V_ENDFILOPN] = 0;   ! Turn off bit
606     0669   4                    IF .ARGS [TST_FMT]               ! Formatted or list-directed
607     0670   4                    THEN
608     0671   4                        CCB [LUB$V_FORMATTED] = 1
609     0672   4                    ELSE
610     0673   5                        BEGIN
611     0674   5                        CCB [LUB$V_UNFORMAT] = 1;
612     0675   5                        CCB [LUB$V_SEGMENTED] = 1;        ! Has to be sequential
613     0676   4                        END;
614     0677   3                    END;
615     0678   3                END
616     0679   2            ELSE IF NOT .ARGS [TST_INT]
617     0680   2            THEN
618     0681   3                BEGIN                     ! Not internal file or ENCODE/DECODE
619     0682   3                L_UNWIND_ACTION = FOR$K_UNWINDRET;
620     0683   3                FOR$$OPEN_DEFLT (
621     0684   3                        !ACCESS = 'SEQUENTIAL' or 'DIRECT'
622     0685   3                        !
623     0686   3                        (IF .ARGS [ARG_REC] THEN OPEN$K_ACC_DIR ELSE OPEN$K_ACC_SEQ),
624     0687   3                        ! TYPE = 'OLD' or 'NEW'
625     0688   3                        !
```

FOR$$IO_BEG   FORTRAN READ/WRITE statement initialization   16-Sep-1984 00:29:21   VAX-11 Bliss-32 V4.0-742   Page 12
2-006   14-Sep-1984 12:32:03   [FORRTL.SRC]FORIOBEG.B32;1   (3)

C 5

```
626   0689  3                          (IF .STMT_TYPE THEN OPEN$K_TYP_NEW ELSE OPEN$K_TYP_OLD),
627   0690  3                          ! FORM = 'FORMATTED' or 'UNFORMATTED'
628   0691  3
629   0692  3                          (IF .ARGS [TST_FMT] THEN OPEN$K_FOR_FOR ELSE OPEN$K_FOR_UNF));
630   0693  3                  L_UNWIND_ACTION = FOR$R_UNWINDPOP;
631   0694  3                  END
632   0695  2              ELSE
633   0696  3                  BEGIN
634   0697  3                  !+
635   0698  3                  ! ENCODE/DECODE or internal file
636   0699  3                  !-
637   0700  3                  CCB [LUB$V_FORMATTED] = 1;
638   0701  3                  CCB [ISB$V_DE_ENCODE] = 1;
639   0702  3
640   0703  3                  IF NOT .ARGS [ARG_USR]    ! Not ENCODE/DECODE?
641   0704  3                  THEN
642   0705  3                      CCB [LUB$A_BUF_PTR] = .UNIT              ! Descriptor is "unit"
643   0706  3                  ELSE
644   0707  4                      BEGIN
645   0708  4                      CCB [LUB$A_BUF_PTR] = RLONG_A (PTR);
646   0709  4                      CCB [LUB$A_BUF_END] = .CCB [LUB$A_BUF_PTR] + .PTR [-3];      ! Length
647   0710  3                      END;
648   0711  3
649   0712  2                  END;
650   0713  2
651   0714  2
652   0715  2          !+
653   0716  2          ! Form local block so we have KEYVAL on stack at JSB time, if
654   0717  2          ! necessary.  It will only be used by UDFO.
655   0718  2          !-
656   0719  2
657   0720  3          BEGIN
658   0721  3
659   0722  3          LOCAL
660   0723  3              KEYVAL;             ! Local copy of ISAM key for conversion between I*2 and I*4
661   0724  3
662   0725  3          !+
663   0726  3          ! Fill in values for ISAM statements.
664   0727  3          ! Normally, this type of thing is done at the REC level, but
665   0728  3          ! why take up space in the ISB when the RAB is already here?
666   0729  3          !-
667   0730  3
668   0731  3          IF .ARGS [ARG_KEY]
669   0732  3          THEN
670   0733  4              BEGIN
671   0734  4
672   0735  4              LOCAL
673   0736  4                  KEY : REF BLOCK [, BYTE];
674   0737  4
675   0738  4              KEY = RLONG_A (PTR);
676   0739  4              CCB [RAB$L_KBF] = .KEY [DSC$A_POINTER];
677   0740  4
678   0741  4              IF .KEY [DSC$W_LENGTH] GTRU 255
679   0742  4              THEN
680   0743  5                  BEGIN
681   0744  5                  FOR$$SIGNAL_STO (FOR$K_INVKEYSPE);
682   0745  5                  RETURN;
```

```
683     0746  4                         END;
684     0747  4
685     0748  4                  !+
686     0749  4                  ! If this is a text string, then use its length.
687     0750  4                  ! If a byte array, treat as a string whose length is the
688     0751  4                  ! array size (for compatibility with PDP-11 FORTRAN IV-PLUS).
689     0752  4                  ! Otherwise, set the key size to zero, which lets RMS use
690     0753  4                  ! whatever key size it wants for numeric values.
691     0754  4                  !-
692     0755  4
693     0756  4                  SELECTONEU .KEY [DSC$B_DTYPE] OF
694     0757  4                      SET
695     0758  4
696     0759  4                      [DSC$K_DTYPE_T] :
697     0760  4                          CCB [RAB$B_KSZ] = .KEY [DSC$W_LENGTH];
698     0761  4
699     0762  4                      [DSC$K_DTYPE_BU, DSC$K_DTYPE_B] :
700     0763  5                          BEGIN
701     0764  5
702     0765  5                          IF .KEY [DSC$B_CLASS] EQLU DSC$K_CLASS_A            ! Byte array
703     0766  5                          THEN
704     0767  6                              BEGIN
705     0768  6
706     0769  6                              IF .KEY [DSC$L_ARSIZE] GTRU 255
707     0770  6                              THEN
708     0771  7                                  BEGIN
709     0772  7                                  FOR$$SIGNAL_STO (FOR$K_INVKEYSPE);
710     0773  7                                  RETURN;
711     0774  6                                  END;
712     0775  6
713     0776  6                              CCB [RAB$B_KSZ] = .KEY [DSC$L_ARSIZE];
714     0777  6                              END
715     0778  5                          ELSE
716     0779  5                              CCB [RAB$B_KSZ] = 0;
717     0780  5
718     0781  4                          END;
719     0782  4
720     0783  4                      [DSC$K_DTYPE_W, DSC$K_DTYPE_WU] :    ! INTEGER*2
721     0784  5                          BEGIN
722     0785  5                          KEYVAL = .(.KEY [DSC$A_POINTER])<0, %BPVAL/2, 1>;        ! Convert word to long
723     0786  5                          CCB [RAB$L_KBF] = KEYVAL;             ! Address of value
724     0787  5                          CCB [RAB$B_KSZ] = 0;                  ! Keysize assumed correct
725     0788  4                          END;
726     0789  4
727     0790  4                      [OTHERWISE] :
728     0791  4                          CCB [RAB$B_KSZ] = 0;                  ! RMS knows the proper key size
729     0792  4                      TES;
730     0793  4
731     0794  4                  !+
732     0795  4                  ! Set KEYID and MATCH parameters.
733     0796  4                  !-
734     0797  4
735     0798  4                  CCB [RAB$V_KGE] = 0;
736     0799  4                  CCB [RAB$V_KGT] = 0;
737     0800  4
738     0801  4                  IF .ARG_LIST_END GEQA .PTR
739     0802  4                  THEN
```

```
; 740      0803   5                      BEGIN
; 741      0804   5                      LOCAL
; 742      0805   5                          KEYID;
; 743      0806   5                      KEYID = RLONG_A (PTR);
; 744      0807   5                      IF .KEYID GEQ 0
; 745      0808   5                      THEN
; 746      0809   5                          IF .KEYID GTR 254
; 747      0810   5                          THEN
; 748      0811   6                              BEGIN
; 749      0812   6                              FOR$$SIGNAL_STO (FOR$K_INVKEYSPE);
; 750      0813   6                              RETURN;
; 751      0814   6                              END
; 752      0815   5                          ELSE
; 753      0816   5                              CCB [RAB$B_KRF] = .KEYID;
; 754      0817   5
; 755      0818   5                      IF .ARG_LIST_END GEQA .PTR
; 756      0819   5                      THEN
; 757      0820   5                          CASE .PTR [0] FROM 0 TO 2 OF
; 758      0821   5                          SET
; 759      0822   5
; 760      0823   5                          [0] :
; 761      0824   5                              ;                              ! Match equal to
; 762      0825   5                          [1] :
; 763      0826   5                              CCB [RAB$V_KGE] = 1;      ! Match greater or equal
; 764      0827   5                          [2] :
; 765      0828   5                              CCB [RAB$V_KGT] = 1;      ! Match greater than
; 766      0829   6                          [OUTRANGE] :
; 767      0830   6                              BEGIN
; 768      0831   6                              FOR$$SIGNAL_STO (FOR$K_INVARGFOR);
; 769      0832   6                              RETURN;
; 770      0833   6                              END;
; 771      0834   5
; 772      0835   5                          TES;
; 773      0836   4                      END;
; 774      0837   4
; 775      0838   3                  END;
; 776      0839   3
; 777      0840   3          !+
; 778      0841   3          ! Call appropriate User data formatted level of abstraction
; 779      0842   3          ! (UDF level = level 2) initialization routine.
; 780      0843   3          !-
; 781      0844   3
; 782      0845   3          JSB_UDFO (FOR$$AA_UDF_PRO + .FOR$$AA_UDF_PRO [.CCB [ISB$B_STTM_TYPE] - ISB$K_FORSTTYLO + 1])
; 783      0846   2          END;                                          ! End of ISAM + JSB
; 784      0847   2
; 785      0848   2          !+
; 786      0849   2          ! Set up I/O in progress handler in caller's frame
; 787      0850   2          !-
; 788      0851   2
; 789      0852   3          BEGIN
; 790      0853   3          LOCAL
; 791      0854   3              FRAME : REF BLOCK [, BYTE];
; 792      0855   3          FRAME = .FP [SF$L_SAVE_FP];                     ! Our caller's frame
; 793      0856   3          CCB [ISB$A_USER_FP] = .FRAME;                   ! Store frame address
; 794      0857   3          CCB [ISB$A_USR_HANDL] = .FRAME [SF$A_HANDLER];       ! Caller's handler
; 795      0858   3          FRAME [SF$A_HANDLER] = FOR$$IO_IN_PROG;         ! Address of I/O in progress handler
; 796      0859   2          END;
```

```
; 797          0860 2
; 798          0861 1     END;                                    ! End of FOR$$IO_BEG routine


;                                                  .TITLE   FOR$$IO_BEG FORTRAN READ/WRITE statement initia
                                                                    lization
                                                   .IDENT   \2-006\

                                                   .PSECT   _FOR$CODE,NOWRT,  SHR,  PIC,2

06  03  04  04  02  02  03  03  04  04  02  02  03  03  00  00000 P.AAA:  .BYTE  0, 3, 3, 2, 2, 4, 4, 3, 3, 2, 2, 4, 4, 3, - ;
                                02  02  03  03  03  03  05  02  0000F                6, 2, 5, 3, 3, 3, 3, 2, 2
29  21  35  35  20  20  02  02  23  23  00  00  21  21  00  00017 P.AAB:  .BYTE  0, 33, 33, 0, 0, 35, 35, 2, 2, 32, 32, - ;
                                30  30  21  21  31  31  08  00  00026                53, 53, 33, 41, 0, 8, 49, 49, 33, 33, 48, - ;
                                                                                   48
8214  C100  C104  C200  C204  0110  0114  0210  0214  0000  0002E P.AAC:  .WORD  0, 532, 528, 276, 272, -15868, -15872, - ;
0214  0000  0000  4110  0114  4210  0214  0000  0000  8210  00042                -16124, -16128, -32236, -32240, 0, 0, - ;
                                      0000  0000  0210  00056                     532, 16912, 276, 16656, 0, 0, 532, 528, - ;
                                                                                  0, 0

                                                   ERR_ADR_IDX=        P.AAA
                                                   STMT_ARG=           P.AAB
                                                   STMT_ATR=           P.AAC
                                                   .EXTRN   FOR$$CB_PUSH, FOR$$ERR_ENDHND
                                                   .EXTRN   FOR$$SIGNAL_STO
                                                   .EXTRN   FOR$$OPEN_DEFLT
                                                   .EXTRN   FOR$$AA_UDF_PRO
                                                   .EXTRN   FOR$$IO_IN_PROG
                                                   .WEAK    FOR$$FMT_COMPIL

                                   083C 00000      .ENTRY   FOR$$IO_BEG, Save R2,R3,R4,R5,R11           : 0347
                      5E       18   C2 00002       SUBL2    #24, SP                                     : 0443
                             08   AE 7C 00005       CLRQ    L_UNWIND_DEPTH
                             10   AE 7C 00008       CLRQ    A_ERR_ADR
                      6D   0252 CF DE 0000B         MOVAL    48$, (FP)
                             53       50 9A 00010   MOVZBL   FLAGS_ARG, STMT_TYPE                       : 0485
                             50       94 00013       CLRB    FLAGS_ARG                                  : 0486
                      55   A2 AF43 9A 00015         MOVZBL   STMT_ARG[STMT_TYPE], ARGS                  : 0487
                      55       50 C8 0001A          BISL2    FLAGS_ARG, ARGS
               14     AE       01 D0 0001D          MOVL     #1, L_UNWIND_ACTION                        : 0497
                      50       6C 9A 00021          MOVZBL   (AP), R0                                   : 0504
               04     AE     6C40 DE 00024          MOVAL    (AP)[R0], ARG_LIST_END
                      50  FF76 CF43 9A 00029        MOVZBL   ERR_ADR_IDX[STMT_TYPE], R0                 : 0505
                      50     6C40 DE 0002F          MOVAL    (AP)[R0], ERR_POS
                      50       04 AE D1 00033       CMPL     ARG_LIST_END, ERR_POS                      : 0506
                             0B       1F 00037       BLSSU    2$
                             05       1B 00039       BLEQU    1$                                        : 0509
               0C     AE     04 A0 D0 0003B         MOVL     4(ERR_POS), A_END_ADR                      : 0511
               10     AE       60 D0 00040 1$:      MOVL     (ERR_POS), A_ERR_ADR                       : 0512
                      55       04 E0 00044 2$:      BBS      #4, ARGS, 3$                               : 0523
                      50       04 CE 00048          MNEGL    #4, R0                                     : 0525
                      52       04 AC D0 0004B        MOVL    UNIT, R2
                             06       11 0004F       BRB      4$
                      50       05 CE 00051 3$:      MNEGL    #5, R0                                     : 0527
                      52       05 CE 00054          MNEGL    #5, R2
               00000000G      00 16 00057 4$:       JSB      FOR$$CB_PUSH
               14     AE       D4 0005D             CLRL     L_UNWIND_ACTION                            : 0529
```

FOR$$IO_BEG
2-006
FORTRAN READ/WRITE statement initialization
G 5
16-Sep-1984 00:29:21   VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:32:03   [FORRTL.SRC]FORIOBEG.B32;1
Page 16
(3)

```
         FF74  CB       10 AE D0 00060              MOVL    A_ERR_ADR, -140(CCB)                    0540
         FF78  CB       0C AE D0 00066              MOVL    A_END_ADR, -136(CCB)                    0541
                     FF70 CB 94 0006C              CLRB    -144(CCB)                              0542
         FF71  CB       53 90 00070              MOVB    STMT_TYPE, -143(CCB)                    0543
               54    FC AB 9E 00075              MOVAB   -4(CCB), R4                            0556
               64 FF54 CF43 B3 00079              BITW    STMT_ATR[STMT_TYPE], (R4)
               6A    13 0007F              BEQL    14$
               52 FF4C CF43 B0 00081              MOVW    STMT_ATR[STMT_TYPE], ATTR              0570
               50    64 B2 00087              MCOMW   (R4), R0                               0571
               52    50 AA 0008A              BICW2   R0, ATTR
   05          52    0E E1 0008D              BBC     #14, ATTR, 5$                          0572
               1A    DD 00091              PUSHL   #26                                    0575
              018E    31 00093              BRW     42$
   05          52    02 E1 00096 5$:          BBC     #2, ATTR, 6$                           0578
               2F    DD 0009A              PUSHL   #47                                    0581
              0185    31 0009C              BRW     42$
   08          52    09 E1 0009F 6$:          BBC     #9, ATTR, 7$                           0595
           0018880C 8F DD 000A3              PUSHL   #1607692
               36    11 000A9              BRB     13$
   09          52    08 E1 000AB 7$:          BBC     #8, ATTR, 8$                           0596
               50 00188804 8F D0 000AF              MOVL    #1607684, R0
               27    11 000B6              BRB     12$
               09    18 000B8 8$:          BGEQ    9$                                     0597
               50 00188814 8F D0 000BA              MOVL    #1607700, R0
               1C    11 000C1              BRB     12$
   16          52    04 E1 000C3 9$:          BBC     #4, ATTR, 11$                          0598
   09          55    03 E1 000C7              BBC     #3, ARGS, 10$                          0599
               50 00188824 8F D0 000CB              MOVL    #1607716, R0
               0B    11 000D2              BRB     12$
               50 0018881C 8F D0 000D4 10$:         MOVL    #1607708, R0
               02    11 000DB              BRB     12$
               50    D4 000DD 11$:         CLRL    R0                                     0598
               50    DD 000DF 12$:         PUSHL   R0                                     0596
               1F    DD 000E1 13$:         PUSHL   #31                                    0591
       00000000G 00    02 FB 000E3              CALLS   #2, FOR$$SIGNAL_STO
               04    00EA              RET
               52    08 AC 9E 000EB 14$:         MOVAB   8(AP), PTR                             0558
   18          55    01 E1 000EF              BBC     #1, ARGS, 17$                          0614
               62    D5 000F3              TSTL    (PTR)                                  0620
               0B    13 000F5              BEQL    15$                                    0623
               E4 AB D5 000F7              TSTL    -28(CCB)
               0B    13 000FA              BEQL    16$                                    0624
         E4  AB       62 D1 000FC              CMPL    (PTR), -28(CCB)
               05    1B 00100              BLEQU   16$
               19    DD 00102 15$:         PUSHL   #25                                    0631
              011D    31 00104              BRW     42$
         E0  AB       82 D0 00107 16$:         MOVL    (PTR)+, -32(CCB)                       0634
               1C    55 E9 0010B 17$:         BLBC    ARGS, 19$                              0645
   07          55    08 E0 0010E              BBS     #8, ARGS, 18$                          0647
         FF7C CB       82 D0 00112              MOVL    (PTR)+, -132(CCB)                      0649
               11    00117              BRB     19$
              FF7C CB 9F 00119 18$:         PUSHAB  -132(CCB)                              0651
              FF72 CB 9F 0011D              PUSHAB  -142(CCB)
               82    DD 00121              PUSHL   (PTR)+
       00000000G 00    03 FB 00123              CALLS   #3, FOR$$FMT_COMPIL
               19    64 E9 0012A 19$:         BLBC    (R4), 21$                              0662
   64     FE  AB       01 E1 0012D              BBC     #1, -2(CCB), 30$                       0665
```

```
                        FE  AB          02  8A 00132            BICB2    #2, -2(CCB)                      : 0668
              06            55          05  E1 00136            BBC      #5, ARGS, 20$                    : 0669
                        01  A4          01  88 0013A            BISB2    #1, 1(R4)                        : 0671
                                        56  11 0013E            BRB      30$
                        01  A4          0A  88 00140 20$:       BISB2    #10, 1(R4)                       : 0675
                                        50  11 00144            BRB      30$                              : 0665
              2D            55          04  E0 00146 21$:       BBS      #4, ARGS, 28$                    : 0679
                        14  AE          02  D0 0014A            MOVL     #2, L_UNWIND_ACTION              : 0682
              04            55          05  E1 0014E            BBC      #5, ARGS, 22$                    : 0692
                                        01  DD 00152            PUSHL    #1
                                        02  11 00154            BRB      23$
                                        02  DD 00156 22$:       PUSHL    #2
                        04              53  E9 00158 23$:       BLBC     STMT_TYPE, 24$
                                        02  DD 0015B            PUSHL    #2
                                        02  11 0015D            BRB      25$                              : 0689
                                        01  DD 0015F 24$:       PUSHL    #1
              04            55          01  E1 00161 25$:       BBC      #1, ARGS, 26$
                                        01  DD 00165            PUSHL    #1                               : 0686
                                        02  11 00167            BRB      27$
                                        02  DD 00169 26$:       PUSHL    #2
         00000000G  00                  03  FB 0016B 27$:       CALLS    #3, FOR$$OPEN_DEFLT
                                14      AE  D4 00172            CLRL     L_UNWIND_ACTION                  : 0693
                                        1F  11 00175            BRB      30$                              : 0679
                        01  A4          01  88 00177 28$:       BISB2    #1, 1(R4)                        : 0700
                        96  AB      40  8F  88 0017B            BISB2    #64, -106(CCB)                   : 0701
              07            55          02  E0 00180            BBS      #2, ARGS, 29$                    : 0703
                        B0  AB      04  AC  D0 00184            MOVL     UNIT, -80(CCB)                   : 0705
                                        0B  11 00189            BRB      30$
                        B0  AB          82  D0 0018B 29$:       MOVL     (PTR)+, -80(CCB)                 : 0708
         B4  AB    B0  AB  F4  A2       C1 0018F            ADDL3    -12(PTR), -80(CCB), -76(CCB)     : 0709
              03            55          03  E0 00196 30$:       BBS      #3, ARGS, 31$                    : 0731
                                      009A  31 0019A            BRW      45$
                        53              82  D0 0019D 31$:       MOVL     (PTR)+, KEY                      : 0738
                        30  AB      04  A3  D0 001A0            MOVL     4(KEY), 48(CCB)                  : 0739
                        00FF  8F          63  B1 001A5            CMPW     (KEY), #255                      : 0741
                                        5E  1A 001AA            BGTRU    38$
                        50          02  A3  9A 001AC            MOVZBL   2(KEY), R0                       : 0756
                        0E              50  91 001B0            CMPB     R0, #14                          : 0759
                                        06  12 001B3            BNEQ     32$
                        34  AB          63  90 001B5            MOVB     (KEY), 52(CCB)                   : 0760
                                        36  11 001B9            BRB      37$
                        02              50  91 001BB 32$:       CMPB     R0, #2                           : 0762
                                        05  13 001BE            BEQL     33$
                        06              50  91 001C0            CMPB     R0, #6
                                        17  12 001C3            BNEQ     34$
                        04          03  A3  91 001C5 33$:       CMPB     3(KEY), #4                       : 0765
                                        23  12 001C9            BNEQ     36$
         000000FF  8F      0C  A3  D1 001CB            CMPL     12(KEY), #255                    : 0769
                                        35  1A 001D3            BGTRU    38$
                        34  AB      0C  A3  90 001D5            MOVB     12(KEY), 52(CCB)                 : 0776
                                        15  11 001DA            BRB      37$                              : 0765
                        03              50  91 001DC 34$:       CMPB     R0, #3                           : 0783
                                        05  13 001DF            BEQL     35$
                        07              50  91 001E1            CMPB     R0, #7
                                        08  12 001E4            BNEQ     36$
                        6E          04  B3  32 001E6 35$:       CVTWL    @4(KEY), KEYVAL                  : 0785
                        30  AB          6E  9E 001EA            MOVAB    KEYVAL, 48(CCB)                  : 0786
```

```
                               34  AB  94 001EE 36$:      CLRB    52(CCB)                                              0791
                   06  AB      60  8F  8A 001F1 37$:      BICB2   #96, 6(CCB)                                          0799
                               52  04  AE  D1 001F6       CMPL    ARG_LIST_END, PTR                                    0801
                                       3B  1F 001FA       BLSSU   45$
                               53      82  D0 001FC       MOVL    (PTR)+, KEYID                                        0806
                                       11  19 001FF       BLSS    40$                                                  0807
              000000FE  8F             53  D1 00201       CMPL    KEYID, #254                                          0809
                                       04  15 00208       BLEQ    39$
                                       31  DD 0020A 38$:   PUSHL   #49                                                 0812
                                       16  11 0020C       BRB     42$
                   35  AB             53  90 0020E 39$:   MOVB    KEYID, 53(CCB)                                       0816
                               52      04  AE  D1 00212 40$:  CMPL    ARG_LIST_END, PTR                                0818
                                       1F  1F 00216       BLSSU   45$
              02             00        62  CF 00218       CASEL   (PTR), #0, #2                                        0820
           0016            0010       001B     0021C 41$:   .WORD   45$-41$,-
                                                                   43$-41$,-
                                                                   44$-41$
                                       30  DD 00222       PUSHL   #48                                                  0831
              00000000G  00        01  FB 00224 42$:   CALLS   #1, FOR$$SIGNAL_STO
                                       04     0022B       RET                                                         0830
                   06  AB             20  88 0022C 43$:   BISB2   #32, 6(CCB)                                          0826
                                       05  11 00230       BRB     45$
                   06  AB      40  8F  88 00232 44$:   BISB2   #64, 6(CCB)                                             0828
                               50  FF71  CB  9A 00237 45$:   MOVZBL  -143(CCB), R0                                     0845
                   50  00000000G0040  D0 0023C       MOVL    FOR$$AA_UDF_PRO[R0], R0
                       00000000G0040  16 00244       JSB     FOR$$AA_UDF_PRO[R0]
                               50      24  AE  D0 0024B       MOVL    36(FP), -FRAME                                   0855
                   FF4C  CB            50  D0 0024F       MOVL    FRAME, -180(CCB)                                     0856
                   FF44  CB            60  D0 00254       MOVL    (FRAME), -188(CCB)                                   0857
                               60  00000000G  00  9E 00259   MOVAB   FOR$$IO_IN_PROG, (FRAME)                          0858
                                       04     00260       RET                                                         0861
                                     0000     00261 46$:   .WORD   Save nothing                                       0443
                               50      08  AC  D0 00263       MOVL    8(AP), R0
                               50      04  A0  D0 00267       MOVL    4(R0), R0
                                       F0  A0  9F 0026B       PUSHAB  L_UNWIND_DEPTH
                                       F4  A0  9F 0026E       PUSHAB  A_END_ADR
                                       F8  A0  9F 00271       PUSHAB  A_ERR_ADR
                                       FC  A0  9F 00274       PUSHAB  L_UNWIND_ACTION
                                       04  DD 00277       PUSHL   #4
                                       5E  DD 00279       PUSHL   SP
                   7E      04  AC  7D 0027B       MOVQ    4(AP), -(SP)
              00000000G  00        03  FB 0027F       CALLS   #3, FOR$$ERR_ENDHND
                                       04     00286       RET
```

; Routine Size: 647 bytes,   Routine Base: _FOR$CODE + 005C

```
; 799      0862 1
; 800      0863 1 END                              ! End of FOR$$IO_BEG module
; 801      0864 1
; 802      0865 0 ELUDOM
```

; J 5

;   PSECT SUMMARY

;   Name   Bytes   Attributes

;   _FOR$CODE   739 NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)

;   Library Statistics

;   -------- Symbols --------   Pages   Processing
;   File   Total   Loaded   Percent   Mapped   Time

;   _255DUA28:[SYSLIB]STARLET.L32;1   9776   18   0   581   00:01.0
;   _255DUA28:[FORRTL.OBJ]FORLIB.L32;1   711   209   29   52   00:00.6
;   _255DUA28:[FORRTL.OBJ]RTLLIB.L32;1   36   0   0   8   00:00.1

;   COMMAND QUALIFIERS

;   BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:FORIOBEG/OBJ=OBJ$:FORIOBEG MSRC$:FORIOBEG/UPDATE=(ENH$:FORIOBEG)

; Size: 647 code + 92 data bytes
; Run Time: 00:18.8
; Elapsed Time: 00:53.7
; Lines/CPU Min: 2760
; Lexemes/CPU-Min: 16088
; Memory Used: 258 pages
; Compilation Complete